JOHN A. O'MALLEY (Bar No. 101181)
FULBRIGHT & JAWORSKI L.L.P.
555 South Flower Street
Forty-First Floor
Los Angeles, California 90071
Telephone: (213) 892-9200
Facsimile: (213) 892-9494
jomalley@fulbright.com

DAN D. DAVISON (*admitted pro hac vice*)
MARC L. DELFLACHE *(admitted pro hac vice)*
FULBRIGHT & JAWORSKI L.L.P.
2200 Ross Avenue, Suite 2800
Dallas, Texas 75201
Telephone: (214) 855-8000
Facsimile: (214) 855-8200
ddavison@fulbright.com
mdelflache@fulbright.com

RICHARD S. ZEMBEK *(admitted pro hac vice)*
FULBRIGHT & JAWORSKI L.L.P.
1301 McKinney, Suite 5100
Houston, Texas 77010
Telephone: (713) 651-5151
Facsimile: (214) 651-5246
rzembek@fulbright.com

Attorneys for Defendants Papa John's USA, Inc.; Wanderspot LLC; Expedia, Inc.; Hotels.com, LP; Hotel Tonight, Inc.; Hotwire, Inc.; Kayak Software Corp.; Orbitz, LLC; Travelocity.com, LP; Fandango, Inc.; StubHub, Inc.; Ticketmaster LLC; Live Nation Entertainment, Inc.; and Micros Systems, Inc.

UNITED STATES DISTRICT COURT

SOUTHERN DISTRICT OF CALIFORNIA

AMERANTH, INC., Plaintiff, v. PIZZA HUT, INC., et al., Defendants.	CASE NOS. 11-cv-1810 JLS (NLS), 12-cv-742 JLS (NLS), 12-cv-739 JLS (NLS), 12-cv-737 JLS (NLS), 12-cv-733 JLS (NLS), 12-cv-732 JLS (NLS), 12-cv-731 JLS (NLS), 12-cv-729 JLS (NLS), 12-cv-858 JLS (NLS), 12-cv-1659 JLS (NLS), 12-cv-1656 JLS (NLS), 12-cv-1655, JLS (NLS), 12-cv-1654 JLS (NLS), 12-cv-1653 JLS (NLS), 12-cv-1652 JLS (NLS), 12-cv-1651 JLS (NLS), 12-cv-1650 JLS (NLS), 12-cv-1649 JLS (NLS), 12-cv-1648 JLS (NLS), 12-cv-1646 JLS (NLS), 12-cv-1644 JLS (NLS), 12-cv-1643 JLS (NLS), 12-cv-1642 JLS (NLS), 12-cv-1640 JLS (NLS), 12-cv-1636 JLS (NLS), 12-cv-1634 JLS (NLS), 12-cv-1633 JLS (NLS), 12-cv-1631 JLS (NLS), 12-cv-1630 JLS (NLS), 12-cv-1629 JLS (NLS), and 12-cv-1627 JLS (NLS))
AND RELATED CASES.	**EXPERT DECLARATION OF RAY R. LARSON**

EXPERT DECLARATION OF RAY R. LARSON

I. INTRODUCTION

1. My name is Ray R. Larson. I am a Professor at the University of California ("UC") at Berkeley's School of Information. I specialize in the design and performance evaluation of information access systems, and the evaluation of user interaction with those systems. My current research focuses on Geographic Information Retrieval, Cross-Language Information Retrieval and Structured (XML) retrieval using probabilistic methods.

2. I received my Ph.D. in Library and Information Studies from UC Berkeley in 1986. My background includes work as a programmer/analyst with the University of California Division of Library Automation (DLA) where I was involved in the design, development, and performance evaluation of the UC public access online union catalog (MELVYL).

3. My research has concentrated on the design and evaluation of information retrieval systems, with an emphasis on digital libraries. I currently teach courses on the design and evaluation of information systems, including IS202 "Information Organization and Retrieval," IS257 "Database Management," IS240 "Principles of Information Retrieval," and IS245 "Organization of Information in Collections."

4. Additional details regarding my qualifications, education, and experience are described in my Curriculum Vitae, which is attached as Exhibit 1.

II. <u>SCOPE OF ASSIGNMENT</u>

5. I have been retained by Defendants Papa John's USA, Inc.; Wanderspot LLC; Expedia, Inc.; Hotels.com, LP; Hotel Tonight, Inc.; Hotwire, Inc.; Kayak Software Corp.; Orbitz, LLC; Travelocity.com, LP; Fandango, Inc.; StubHub, Inc.; Ticketmaster LLC; Live Nation Entertainment, Inc.; and Micros Systems, Inc. in the above-captioned cases and submit this Expert Declaration.

6. I am being compensated for my time at my customary rate of $350 per hour plus any out-of pocket expenses. I am being paid for my time, regardless of the facts I know or discover and regardless of the conclusions or opinions that I reach and express. I have no financial interest in the outcome of this case.

7. I have been asked to review the following:

a. U.S. Patent No. 6,384,850, to McNally et al., entitled "Information Management and Synchronous Communications System with Menu Generation;"

b. U.S. Patent No. 6,871,325, to McNally et al., entitled "Information Management and Synchronous Communications System with Menu Generation;"

c. U.S. Patent No. 8,146,077, to McNally et al., entitled "Information Management and Synchronous Communications System with Menu Generation, and Handwriting and Voice Modification of Orders;"

d. Lydia Pallas Loren & Andy Johnson-Laird, Computer Software-Related Litigation: Discovery and the Overly-Protective Order, Vol. 6, Issue 1, Federal Courts Law Review (2012) ("Loren & Johnson-Laird");

e. Ameranth's Original Complaints against (i) Papa John's; (ii) GrubHub; (iii) Wanderspot; (iv) Starwood; (v) Orbitz; (vi) StubHub; and (vii) Micros (collectively, "Original Complaints").

f. Plaintiff Ameranth's Technology Tutorial Presentation; and

g. Defendants' Technology Tutorial Presentation.

8. I have been asked to provide my opinions regarding:

a. the categories and/or types of source code and other components involved in a typical e-commerce system, such as those described in the Original Complaints;

b. the challenges associated with collecting and producing the "entire source code tree" and all of the ancillary control files, as well as all of the third party components that are necessary for the accused systems to function (i.e., a "full build environment");

c. the relevance of source code to the claims of the patents-in-suit; and

d. whether an expert or consultant would need or want access to the "entire source code tree" for a complex e-commerce system in order to analyze specific functionality.

9. My opinions regarding the issues that I have been asked to consider appear below.

III. SUMMARY OF OPINIONS

10. The source code production scheme advocated by Loren & Johnson, which calls for production of the "entire source code tree" and any other files and components necessary to create a working version of the accused program (i.e., a "full build environment"), appears to focus on a scenario in which a single software program runs on a single machine that uses a single operating system. An e-commerce system is markedly different from the basic system contemplated by Loren & Johnson-Laird because of the complexity and different types of source code, computer languages, hardware systems, operating systems, third party interfaces, applications, and data (much of which may remain under the third party's control) that are involved in an e-commerce system. As a result, producing a "full build environment" for many of the accused e-commerce systems is potentially a monumental task, to the point of being extremely burdensome from a labor and cost perspective, if not impossible. By my estimation, the labor required to collect and produce all such materials and set up a working version of an accused system, if it can even be done at all, could be well over one hundred man-hours and the cost (in the form of hardware and third party software licenses that must be acquired), could be well over $100,000.

11. An expert or consultant analyzing source code for a patent case does not need access to the "entire source code tree" for an accused system because there is nothing to be gained (other than significantly increased consulting fees) from analyzing source code that has no relevance to the system aspects that are alleged to embody the elements of the asserted patent claims. In my opinion, most of the source code for the accused systems is unlikely to have any relevance to the claims of the patents-in-suit. Therefore, rather than allow the plaintiff's experts or consultants to review the "entire source code tree" to independently determine which parts of each defendant's source code are relevant to the asserted claims of the patent, in my opinion it would be much more efficient to have the defendant's technical employees, who are intimately familiar with their own source code, identify and provide the portions of source code that

specifically relate to the aspects or functionality identified by the plaintiff in its infringement contentions.

IV. <u>DETAILED DISCUSSION</u>

A. <u>Source Code Generally</u>

12. Source code is a collection of computer instructions written in a human-readable computer language, usually as text. Typically, a program called a compiler is used to translate the source code into object code, which is essentially machine instructions and data, that a computer can directly read and execute. The object code is then stored as executable files that can be executed by a computer. Most computer applications are distributed in a form that includes only the executable files and not the underlying source code, which is usually guarded as highly confidential intellectual property. Alternatively, some programming languages, known as interpreted languages (such as Python), use a program called an interpreter to both translate the program text and execute the translated machine instructions. Virtually all commercially available programs are distributed in object code form, and not as interpreted code.

B. <u>The Basic System of Loren & Johnson-Laird</u>

13. Loren & Johnson-Laird state: "Given the huge size of modern programs, the most technically viable way of determining completeness is to compile the source code into a working program. This task demands all of the source code and all of the ancillary control files, as well as all of the third party components (be they source or object code) that are necessary for the program to function." II.B.1. This discussion appears to focus on the production of source code for a single software program that runs on a single machine that uses a single operating system. With such a basic system, producing the materials necessary for the receiving party to compile the source code into a working program is usually not particularly difficult, in part because (1) the source code for the single software program is often maintained using a single Source Code Control System (SCCS), and (2) no special hardware or third-party software is needed to execute the software after it is compiled onto a machine that uses the correct operating system.

C. E-Commerce Systems

14. The e-commerce systems described in the Original Complaints involve applications running on iPhones and Android mobile phones, client devices (e.g., computers, tablets, and smart phones) using web browsers to communicate through the Internet to web servers, and web servers that communicate to one or more application, database, and third party servers.

15. To illustrate the complexity of these systems and their operation, consider the following. A web page can be defined as "A document on the World Wide Web. A Web page consists of an HTML file, which associated files for graphics and scripts, in a particular directory on a particular machine (and thus identifiable by a URL)." Microsoft Computer Dictionary at 564. In the case of static content, a user can request a particular web page by typing a URL into her browser. For example, if the user typed in http://www.casd.uscourts.gov/addressChanges.html, her browser would break the URL into four parts: (1) protocol (http), (2) domain (uscourts.gov), (3) subdomain (casd), and (4) requested file (addressChanges.html). The browser would communicate through the Internet with a name server to obtain an Internet Protocol (IP) address for www.casd.uscourts.gov/. The browser then uses the IP address to route a GET request for the static document addressChanges.html to the web server. The web server then sends the HTML text for the document to the browser, which reads and renders it for display on the client's screen, which often involves resolving and fetching additional files for images or other elements of the page included by reference using additional URLs.

16. In an e-commerce system (which typically uses very little static content), the process is arguably similar but far more complex, and the process typically involves multiple servers and systems. A typical e-commerce system will likely include, by way of example only, search and navigation functionality that allows customers to search and browse for items or services available for purchase, purchasing and payment systems functionality (usually incorporating encryption technology) that allows customers to complete the purchase of their selected items and services, and order tracking/fulfillment functionality that allows customers to provide shipping information and track the status of their orders.

17. All of this varied functionality may be developed and maintained by different groups within a company and typically involves different types of source code, computer languages, hardware systems, operating systems, third party interfaces, applications, and data (much of which may remain under the third party's control). Software platforms used by an e-commerce system can include Java, ColdFusion, Active MQ, staged event-driven architecture (SEDA), Lucene/Solr, JBoss, XSL, Oracle, Teamworks, Apache HttpClient, and Apache Tapestry, among many others, many of which require specialized hardware and operating systems in order to function. Depending on the functionality utilized by a customer interacting with the e-commerce system, some or all of these platforms may be invoked to provide the functionality, and additional functionality is commonly provided to support user interaction that is part of the web pages transmitted to the user, and executed by the users' own web browser using JavaScript. Because of the complex and distributed nature of e-commerce systems, the "source code tree" for such systems will likely not exist in a single tree or make use of a unified SCCS. More commonly, to provide the varying types of functionality, the operational systems consist of multiple programs, each operating in a distinct operating system environment on a distinct server and interacting with internal and external programs and data.

18. In many instances, e-commerce systems such as those identified in the Original Complaints have a combination of proprietary software and data and third party software and data that may be accessed over the Internet or secure networks. For example, the search and navigation functionality for an event ticketing website may rely on integration with a third party concert venue's inventory system, the purchasing and payment systems functionality may rely on integration with a third party bank or credit card issuing institution's payment system, and the order tracking/fulfillment functionality may rely on integration with a third party shipping vendor's tracking system.

D. **<u>The Challenges with Producing a "Full Build Environment" for an E-Commerce System</u>**

19. As should be evident from the foregoing discussion, producing all of the source code (which Loren & Johnson-Laird refer to as the "entire source code tree") and any other files and components necessary to create a working version of an e-commerce system (i.e., a "full build environment") is potentially a monumental task.

20. First, all of the source code for all of the system's functionality must be collected. Depending on the amount and complexity of the system's functionality, the "entire source code tree" may be comprised of several million of lines of code broken into thousands of different functional modules maintained by tens of different software development teams spread around the world (with each development team being responsible for writing and maintaining the source code for a different segment of the system's functionality). Thus, simply collecting all of the source code could require weeks or even months to coordinate.

21. Of course, for any third-party software that is utilized to provide certain functionality in the e-commerce system, collecting the source code will likely not be possible, as that source code will not be in the possession of the e-commerce system owner, nor will the vendor of the third-party software be willing to voluntarily share its source code. Thus, to create the "full build environment," the only option in most instances will be to produce an executable copy of any necessary third-party software. Loren & Johnson-Laird seem to recognize this

possibility in their statement that the producing party should produce "all of the third party components (be they source or object code) that are necessary for the program to function." II.B.1. The problem with producing executable copies of third-party software is that additional licenses from the vendors of that software must be purchased for the produced copies, which could cost on the order of tens of thousands of dollars. Loren & Johnson-Laird acknowledge this problem but then dismiss it with a bizarre "hope for the best" strategy: "A party may assert that such production [of third party software without acquiring the proper licenses] would be a breach of the license agreement for that software or that such production would constitute copyright infringement. To the authors' knowledge there has never been a breach of license or copyright action filed when third party source code and/or software has been produced in the context of litigation and subject to a protective order." III.B.2. Loren & Johnson-Laird seem unaware of the many commercial software packages that actively validate their licenses at run time, which are typically keyed to particular machine identifiers. Such packages will not operate at all without a valid license for a given machine.

22. Next, in order to create the "full build environment" for the e-commerce system, the producing party would need to acquire and produce the relevant hardware on which the various software and database components of the system are designed to operate. This may include specialized types of content servers, web servers, and modified legacy hardware upon which the system is designed to operate, as well as special-purpose mobile devices with synchronous radio communications. Such hardware could cost tens—or perhaps even hundreds—of thousands of dollars to acquire, assuming it is even possible to acquire certain hardware (which may not be the case for legacy hardware that is no longer sold).

23. Assuming all of the relevant source code, third party software, and hardware necessary to create the "full build environment" can be acquired and produced, there will still be an issue with acquiring access to any third party systems that supply functionality, content, or data to the e-commerce system (e.g., a third party airline ticket inventory system, a third party payment system, or a third party shipping system). It is highly unlikely that those third parties

will allow a "dummy" version of the e-commerce system that has been re-created for litigation purposes to interface with and pull data from their third party systems. And without access to those third party systems, it will be impossible to create a "working version" of the accused e-commerce systems, as advocated by Loren & Johnson-Laird.

24. The final challenge with producing a "full build environment" for the accused systems is that even if all of the relevant source code, third party software, and hardware necessary to create the "full build environment" can be acquired and produced and access to the necessary third party systems can be acquired, it will likely be impossible for the receiving party to actually set up a "working version" of the accused system without significant assistance from one or more employees of the producing party who have the requisite knowledge of how the system is constructed. Thus, it will likely be necessary for one or more employees of the producing party to travel to the site where the "full build environment" is to be constructed and spend several weeks or even months setting up the system. And of course, once the system is set up and connected to the Internet for full operation, the producing party's source code will then be at a higher risk of disclosure. Loren & Johnson-Laird suggest that such a system would be completely isolated from the internet, which actually adds several layers of complexity in setting up the functionality of the system, since a "private internet" must be constructed to support the system and all of its dependent components running on internet connected machines.

E. <u>The Uselessness of Analyzing the "Entire Source Code Tree"</u>

25. An expert or consultant analyzing source code for a patent case does not need access to the "entire source code tree" for an accused system, as there is nothing to be gained (other than significantly increased consulting fees) from analyzing source code that has no relevance to the aspects or functionality that are alleged to embody or perform the elements of the asserted patent claims. Indeed, Loren & Johnson-Laird confirm my opinion by acknowledging that much of the source code for an accused system will have "no relevance" and will be "useless" in a patent case: "Much of the source code that the expert has to analyze will

have no relevance to the claims of the patent asserted. . . . It is quite usual for a significant portion of the actual forensic software analysis to be useless . . . ." II.A.2.

26. Typically, at least in well-designed systems, particular functionality is embodied within a relatively small number of modules and verification of the behavior of those modules is often obvious in isolation. There may also be cases where observed behavior of a system may involve multiple elements of the system, but only in the simplest systems (usually single programs) is the full source code tree implicated in any given functionality. As I have already suggested above, e-commerce systems are not simple single programs, but complex systems where typically only a small fraction of the full functionality of the system is at issue for any given patent claim. Based on my review of the claims of the patents-in-suit and my understanding of complex e-commerce systems as discussed above, it is my opinion that most of the source code for the accused systems is unlikely to have any relevance to the claims of the patents-in-suit.

27. Given the admitted uselessness of a "significant portion" of the source code in patent infringement cases, it is curious that Loren & Johnson-Laird nevertheless advocate for the production of the "entire source code tree" in such cases. Their rationale appears to be that "only hindsight can determine which were those parts of the source code relevant to the asserted claims of the patent." II.A.2. In my experience, it is completely unnecessary for a party to have their expert or consultant review the "entire source code tree" to independently determine which parts of the source code are relevant to certain functionality or to the asserted claims of the patent. In the patent litigation context, it is much more efficient to have the defendant's technical employees, who are intimately familiar with their own source code, identify and provide the portions of source code that specifically relate to the aspects or functionality identified by the plaintiff in its infringement contentions. If an expert or consultant has questions after reviewing the produced source code, these questions are more efficiently addressed and answered through questioning of the defendant's technical employees via interrogatories, requests for admission, or deposition questions that an expert or consultant can assist counsel in formulating.

28. The potential for waste is particularly apparent in this case, where there are over 30 different defendants. It would border on impossible for the plaintiff in this case to have 2-3 experts meaningfully review an "entire source code tree" for each of the 30+ defendants and independently determine, in a reasonably short amount of time, which portions of that code specifically relate to the accused aspects or functionality. Given the volume of code that likely comprises the "entire source code tree" for each defendant, it might take the plaintiff's experts several months per defendant to review all of the code, thereby causing significant delay in the case. Even if a consultant or expert could identify the germane source code modules, he or she would likely want counsel to confirm his or her understanding through interrogatories, requests for admission, or technical deposition questions that the expert or consultant routinely formulates. That is, having the experts or consultants review the "entire source code tree" would unnecessarily add an additional step in the process, with significant potential for increased expert fees and delay.

I declare under penalty of perjury under the laws of the United States of America that the foregoing is true and correct.

Executed on January 10th_, 2013, at Wailua, Hawaii.

[illegible]

RAY R. LARSON